Modelica Overview

by Martin Otter

Abstract:
This slide set gives an overview about the Modelica language, including users view, libraries and a sketch of the language elements.
### License

This slide set is provided "as is" without any warranty. It is licensed under the CC-BY-SA (Creative Commons Attribution-Sharealike 3.0 Unported) License (= the license used by Wikipedia). Human-readable summary of the license text:

You are free:

- **to Share** — to copy, distribute and transmit the work, and
- **to Remix** — to adapt the work

Under the following conditions:

- **Attribution** — You must attribute the work in the manner specified by the author or licensor (but not in any way that suggests that they endorse you or your use of the work.)
- **Share Alike** — If you alter, transform, or build upon this work, you may distribute the resulting work only under the same, similar or a compatible license.

The legal license text and disclaimer is available from here: [http://creativecommons.org/licenses/by-sa/3.0/legalcode](http://creativecommons.org/licenses/by-sa/3.0/legalcode)

### Revisions:

<table>
<thead>
<tr>
<th>Date</th>
<th>Author/Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>2009-07-17</td>
<td>Martin Otter (DLR-RM and Chairman of Modelica Association): First version, based on material from courses given at Technical University of Munich.</td>
</tr>
<tr>
<td>2013-08-28</td>
<td>Dietmar Winkler( Telemark University College)</td>
</tr>
<tr>
<td></td>
<td>Updated information on MSL and MA</td>
</tr>
</tbody>
</table>

---

[Modelica Overview, Slide 2]
Contents

1. Modelica Introduction
2. Modelica Users View
3. Modelica Libraries
4. Modelica Language Elements
1. Modelica Introduction

Goal of Modelica:

- Modeling the **dynamic behavior** of **technical systems** consisting of components from, e.g., mechanical, electrical, thermal, hydraulic, pneumatic, fluid, control and other domains in a **convenient way**.
- Models are described by **differential, algebraic, and discrete equations**.
- No description by partial differential equations, i.e., no FEM (finite element method) and no CFD (computational fluid dynamics), but using results of, e.g., FEM programs.
- Modelica is used in industry since year 2000.
Example: *detailed vehicle model*

- **Vehicle dynamics** (3-dim. mechanics)
- **Drive trains** (1-dim. mechanics)
- **Hydraulics**
- **Combustion**
- **Air Conditioning** (Thermofluid systems)
- **Electrical/electronic systems**
- **Electrical machines**
- **Hierarchical state machines**
- **Control** (Input/output blocks, ...)

(courtesy: Modelon AB)
Modelica Language und Simulation-Environments

Graphical editor for Modelica models

Modelica simulation environment (free or commercial)

Textual description on file (equations, "schematic", animation)

Free Modelica language

Translation of Modelica models in C-Code, Simulation, and interactive scripting (plot, freq. resp., ...)

Modelica Simulation-environment (free or commercial)
Commercial Modelica Simulation Environments (alphabetical list)

• **CATIA Systems** from Dassault Systèmes (based on Dymola kernel with PLM integration)

• **CyModelica** from CyDesign

• **Dymola** from Dynasim AB, Sweden (Dynasim was acquired by Dassault Systèmes in 2006).

• **LMS Imagine.Lab AMESim** from LMS International

• **MapleSim** from MapleSoft, Canada.

• **MathModelica** from Wolfram Research, Sweden.

• **SimulationX** from ITI GmbH, Dresden, Germany.

Free Modelica Simulation Environments (alphabetical list)

• **JModelica.org** from Lund University and Modelon AB, Sweden (under development; subset of Modelica is available).

• **OpenModelica** from Linköping University, Sweden (under development; subset of Modelica is available)

An up-to-date list of Modelica tools is available from [www.modelica.org/tools](http://www.modelica.org/tools)
Modelica Association

- **Modelica** is a **free language** and is developed by the (non-profit) Modelica Association since 1996:
  - 2000: First applications
    ...
  - 2005: **Modelica 2.2**
  - 2007: **Modelica 3.0**
    ...
  - 2012: **Modelica 3.3** (current)
- Develops also the largest, free library for multi-domain models (**Modelica Standard Library**)
- 112 "individual" and 16 "organizational members" (interested in "active" individual members; Therefore requirement: participation at 2 Modelica Design Meetings in the last 12 months).
- 9 International Modelica Conferences (Modelica'2012 with 400 participants)
- All infos under [http://www.modelica.org](http://www.modelica.org) (Specification, simulation environments, free libraries, 400 papers, ...)

66th Design Meeting in Hamburg, March 2010
(after release of Modelica 3.2)
2. Modelica Users View

- Each **icon** represents a **physical component**.
  (electrical resistance, mechanical device, pump, ...)

- A **connection line** represents the actual physical **coupling** (wire, fluid flow, heat flow, ...)

- A component consists of **connected** sub-components (= hierarchical structure) and/or is described by **equations**.

- By **symbolic** algorithms, the high level Modelica description $0 = f(\dot{x}(t), x(t), y(t), t)$ is transformed into a set of explicit differential equations:
  $$\dot{x}(t) = f(x(t), t)$$
  $$y(t) = f(x(t), t)$$

1000 non-trivial algebraic equations, 80 states. Faster as real-time on slow PC.

model Resistor
extends OnePort;
 parameter Real R;
 equation
  v = R*i;
end Resistor;
Example: Hardware-in-the-Loop Simulation of automatic gear boxes
(different vehicle manufacturers)

Electronic Control Unit
(Hardware)

desired pressure

(Simulation)

+ driver + engine
+ 1D vehicle dynamics
3. Modelica Libraries

Library „Modelica“ is the Modelica Standard Library which is developed from the Modelica Association. It is freely available in source code and can be modified and used in commercial programs.

Continuous development since 1998.
Newest version 3.2.1 from August 2013:

1340 generic models
1000 functions
1450 packages (mostly media definitions)
Analog electric and electronic components, such as resistor, capacitor, transformers, diodes, transistors, transmission lines, switches, sources, sensors.

Digital electrical components based on the VHDL standard, like basic logic blocks with 9-valued logic, delays, gates, sources, converters between 2-, 3-, 4-, and 9-valued logic.

Electrical machines (uncontrolled asynchronous-, synchronous-, DC-machines)

Simple thermo-fluid pipe flow, especially to model cooling of machines with air or water (pipes, pumps, valves, ambient, sensors, sources) and lumped heat transfer with heat capacitors, thermal conductors, convection, body radiation, sources and sensors.
**Library Modelica: Media and Mechanical Libraries**

- **Large media library** with
  - 1240 gases and mixtures between these gases.
  - table based media ($h = h(T)$, etc.)
  - high precision model for water (IF97)
  - moist air.

1-dim. mechanical systems, e.g., drive trains, planetary gears, convenient definition of speed/torque dependent friction (clutches, brakes, bearings, ..)

3-dim. mechanical systems consisting of joints, bodies, force and sensor elements. Joints can be driven by drive trains defined by 1-dim. mechanical system library.
**Library Modelica: Control and Script Libraries**

Continuous and discrete input/output blocks, e.g., PI, PID, transfer function, state space, filter, logical, non-linear, routing, table source blocks

Hierarchical state machines with same modeling power as Statecharts. Modelica is used as synchronous action language, i.e. deterministic behavior is guaranteed (not the case for Statecharts)

Logical blocks such as "and, or, edge, timer, ", ...

\[
A = \begin{bmatrix}
1 & 2 & 3 \\
3 & 4 & 5 \\
2 & 1 & 4
\end{bmatrix};
\]
\[
b = \{10, 22, 12\};
\]
\[
x = \text{Matrices.solve}(A, b);
\]
\[
\text{Matrices.eigenValues}(A);
\]

Functions on matrices, such as for solving linear systems, eigen and singular values etc.,

and functions operating on strings, streams, files, e.g., to copy and remove a file or sort a vector of strings.
Library Modelica: Sublibraries that were added in 3.1

**Electro-magnetic** devices with lumped magnetic networks. E.g. flux tubes, magnetic sources and sensors, magnetic materials.

General library for **fluid pipe flow** for all media of Modelica.Media
- one and multiple substances
- one and multiple (homogenous) phases
- incompressible and compressible

### Standard conform libraries developed by the MA

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Last Release</th>
<th>Last Active</th>
</tr>
</thead>
<tbody>
<tr>
<td>ModelicaStandardLibrary</td>
<td>Free (standard conform) library from the Modelica Association to model mechanical (1D/3D), electrical (analog, digital, machines), thermal, fluid, control systems and hierarchical state machines. Also numerical functions and functions for strings, files and streams are included.</td>
<td>v3.2.1+build.2(6 days ago)</td>
<td>6 days ago</td>
</tr>
<tr>
<td>Modelica_DeviceDrivers</td>
<td>Free (standard conform) library for interfacing hardware drivers to Modelica models. There is support for joysticks, keyboards, UDP, shared memory, AD/DA converters and other devices.</td>
<td>v1.1build2(3 months ago)</td>
<td>23 days ago</td>
</tr>
<tr>
<td>Modelica_Synchronous</td>
<td>Free (standard conform) library to precisely define and synchronize sampled data systems with different sampling rates. It provides convenient to use blocks to utilize the new synchronous language elements introduced in Modelica 3.3.</td>
<td>v0.91(11 months ago)</td>
<td>3 months ago</td>
</tr>
</tbody>
</table>

### Other libraries developed by the MA

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Last Release</th>
<th>Last Active</th>
</tr>
</thead>
<tbody>
<tr>
<td>ExternalMedia</td>
<td>The ExternalMedia library provides a framework for interfacing external codes computing fluid properties to Modelica.Media-compatible component models.</td>
<td>N/A</td>
<td>a month ago</td>
</tr>
<tr>
<td>Modelica_EnergyStorages</td>
<td>Free library that contains models with different complexity for simulating of electric energy storages like batteries (single cells as well as stacks) interacting with loads, battery management systems, loads and charging devices.</td>
<td>N/A</td>
<td>5 months ago</td>
</tr>
<tr>
<td>Modelica_LinearSystems2</td>
<td>Free library providing different representations of linear, time invariant differential and difference equation systems, as well as typical operations on these system descriptions.</td>
<td>v2.3 (a year ago)</td>
<td>5 months ago</td>
</tr>
<tr>
<td>Modelica_StateGraph2</td>
<td>Free library providing components to model discrete event, reactive and hybrid systems in a convenient way with deterministic hierarchical state diagrams. Modelica_StateGraph2 is not fully Modelica compliant and will never be, since a better solution is now available with Modelica 3.3</td>
<td>v2.0.1 (3 years ago)</td>
<td>5 months ago</td>
</tr>
<tr>
<td>PowerSystems</td>
<td>The library is intended to model electrical power systems at different levels of detail both in transient and steady-state mode.</td>
<td>v0.2 (4 months ago)</td>
<td>23 days ago</td>
</tr>
</tbody>
</table>
Quickly growing number of commercial libraries. Small selection:

**SmartElectricDrives (ATI, Austria)**
Controlled electrical machines with quasi-stationary and transient models, e.g.,
controllers (voltage/frequency, field-oriented, speed/position),
power electronics (AD/DC, DC/AC, DC/DC converters, PWM),
energy storages (batteries, supercaps, fuel cells), ...

---

**Hydraulic/Pneumatic Libraries (Modelon AB, Sweden)**
Libraries to model pipe networks for oil and air. Contain all important standard components like pumps, valves, volumes, lines, sensors

---

**PowerTrain (DLR-RM, Germany)**
Library to model vehicle power trains and all type of planetary gearboxes. E.g. standard and planetary gears with losses, clutches with friction, flexible driveline models, automatic gearboxes, optional 3D effects (mounting on vehicle)
4. Modelica Language Elements

Example: Definition of Capacitor

```modelica
connector Pin
    Voltage v;     // identical at connection
    flow Current i; // sums to zero at connection
end Pin;

partial model TwoPin
    Pin p, n;   Voltage v;
    equation
        v = p.v - n.v;
        0 = p.i + n.i;
end TwoPin;

model Capacitor
    extends TwoPin;
    parameter Capacitance C;
    equation
        C*der(v) = p.i;
end Capacitor;
```

Example: Hierarchical Modelica Model

```model
MotorDrive =
  PI controller;
  Ramp ramp;
  Motor motor;
  Gearbox gear(ratio = 100);
  Inertia inertia(J = 10);
  SpeedSensor tacho;

equation
  connect(controller.y, motor.i_ref);
  connect(motor.flange, gearbox.flange_a);
  connect(gearbox.flange_b, inertia.flange_a);
  connect(inertia.flange_b, tacho.flange);
  connect(tacho.w, controller.u_m);
  connect(ramp.y, controller.u_r);
end MotorDrive;
```

This slide demonstrates the implementation of a hierarchical Modelica model for a motor drive system. The textual representation shows the model declaration and the connections between the components of the system. The graphical representation complements the textual model with visual elements, illustrating how the components are connected and how the system behaves over time. The diagram includes a generic model representation with placeholders for specific values and connections, which are elaborated in the textual model to provide a complete description of the system's operation.
Many pre-defined connectors, e.g.:

- input/output signals
- electrical pins
- electrical plugs (multi-phase networks)
- VHDL digital input/output signals
- 3-dim. mechanical frame
  - 1-dim. rotational mechanical flange
  - 1-dim. translational mechanical flange
- 1-dim. heat transfer
- fluid port (for all media from Modelica.Media)
- signal bus
### Different types of variables in a connector definition

<table>
<thead>
<tr>
<th>Category</th>
<th>Example</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>input/output</strong></td>
<td><strong>input</strong> Real u</td>
<td>Connected variables are identical; block diagram connection restrictions</td>
</tr>
<tr>
<td><strong>potential</strong></td>
<td>Real v;</td>
<td>Connected variables are identical</td>
</tr>
<tr>
<td><strong>flow</strong></td>
<td><strong>flow</strong> Real i;</td>
<td>Sum of the connected variables is zero</td>
</tr>
<tr>
<td><strong>stream</strong></td>
<td><strong>stream</strong> Real h;</td>
<td>Describes bi-directional flow of matter (more complicated definition)</td>
</tr>
<tr>
<td><strong>overdetermined</strong></td>
<td></td>
<td>Redundant set of variables with constraint equations, e.g., orientation matrix, dq0 transformation (more complicated def.)</td>
</tr>
<tr>
<td><strong>signal bus</strong></td>
<td><strong>explandable</strong></td>
<td>Content defined by signals connected to the connector.</td>
</tr>
<tr>
<td><strong>connector</strong></td>
<td><strong>end</strong> Bus;</td>
<td></td>
</tr>
</tbody>
</table>
Other Language Elements

- Mathematical notation for **matrices** and **arrays**
- **Arrays** not only from numbers but also **from models** (e.g. arrays of resistors).
- **Replaceable submodels**, e.g., to change quickly between different versions of a transmission in a vehicle system model.
- Language elements to define conveniently **discontinuous** and **variable structure** systems, e.g., to model friction or ideal switches.
- Mathematical **functions** with varying number of input/output arguments. The procedural part of Modelica is used as scripting language.
- Convenient calling of **C, Fortran**, and **Java** functions within Modelica.
- **Powerful library concept**
  (Modelica tool has enough information to find model in the file system automatically, version handling, transformations between versions, ...).